Sharp Noisy Binary Search with Monotonic Probabilities

Lucas Gretta
University of California, Berkeley, CA, USA

Eric Price
University of Texas at Austin, TX, USA

Abstract

We revisit the noisy binary search model of [10], in which we have n coins with unknown probabilities p_i that we can flip. The coins are sorted by increasing p_i, and we would like to find where the probability crosses (to within ε) of a target value τ. This generalized the fixed-noise model of [2], in which $p_i = \frac{1}{2} \pm \varepsilon$, to a setting where coins near the target may be indistinguishable from it. It was shown in [10] that $\Theta(\frac{1}{\varepsilon^2} \log n)$ samples are necessary and sufficient for this task.

We produce a *practical* algorithm by solving two theoretical challenges: high-probability behavior and sharp constants. We give an algorithm that succeeds with probability $1 - \delta$ from

$$\frac{1}{C_{\tau,\varepsilon}} \cdot \left(\log_2 n + O(\log^{2/3} n \log^{1/3} \frac{1}{\delta} + \log \frac{1}{\delta})\right)$$

samples, where $C_{\tau,\varepsilon}$ is the optimal such constant achievable. For $\delta > n^{-o(1)}$ this is within $1 + o(1)$ of optimal, and for $\delta \ll 1$ it is the first bound within constant factors of optimal.

2012 ACM Subject Classification Theory of computation → Randomness, geometry and discrete structures; Theory of computation → Streaming, sublinear and near linear time algorithms; Theory of computation → Lower bounds and information complexity

Keywords and phrases fine-grained algorithms, randomized/probabilistic methods, sublinear/streaming algorithms, noisy binary search

Digital Object Identifier 10.4230/LIPIcs.ICALP.2024.75

Category Track A: Algorithms, Complexity and Games

Related Version *Full Version*: https://arxiv.org/abs/2311.00840

Funding *Lucas Gretta*: NSF Award CCF-2231095. Work partially done while at UT Austin.
Eric Price: NSF awards CCF-2008868, CCF-1751040 (CAREER), and the NSF AI Institute for Foundations of Machine Learning (IFML).

1 Introduction

Binary search is one of the most fundamental algorithms in computer science, finding an index $i^* \in [n]$ from $\log_2 n$ queries asking if a given index i is larger than i^*. But what if the queries are noisy?

One model for noisy binary search has each query be incorrect independently with exactly the same probability $\frac{1}{2} - \varepsilon$. In this model, which we call FixedNoiseNBS, a line of work [2, 1, 5, 9] has found a sharp bound for the required expected sample complexity, with tight constants. However, in many applications of noisy binary search the error probability is not fixed, but varies with i: comparing i to i^* is much harder when i is close to i^*.

As one example, consider the problem of estimating the sample complexity of an algorithm such as for distribution testing or noisy binary search itself. Proofs in this space are often sloppy with constant factors, so the proven bound is not reflective of the true performance. If so, we would like to empirically estimate the sample complexity i at which the success

Table 1 Comparison of our result to prior algorithms for MonotonicNBS in the regime of $\varepsilon \ll \min(\tau, 1-\tau)$ and $\delta = 1/n^{o(1)}$, ignoring lower order terms. The analysis in [10] is not careful with constants, so we also include our best estimate of the actual constant after tuning constant factors in the algorithms.

Algorithm	Proven query complexity	Actual constant
Binary Search w/ Repetition	$2\frac{\tau(1-\tau)}{\varepsilon^2} \log n \cdot \log \frac{\log_2 n}{\delta}$	
[10] Multiplicative Weights	$4000\frac{\max(\tau,1-\tau)}{\varepsilon^2} \log n \cdot \log \frac{1}{\delta}$	≈ 31
[10] Backtracking	$476909\frac{\max(\tau,1-\tau)}{\varepsilon^2} \log n \cdot \log \frac{1}{\delta}$	≈ 2000
BayesianScreeningSearch	$2\frac{\tau(1-\tau)}{\varepsilon^2} \log n$	

probability p_i is above a given threshold τ (say, 90%). (In some cases we even know the worst-case distribution [6] so the empirical estimate is of the worst-case performance, not just the distributional performance.) We can run the algorithm at a given sample complexity i and check correctness, getting Success with probability p_i. The success probability is monotonic in i, and we would like to estimate the i^* where p_i crosses τ. Finding i^* exactly may be very hard – the success probability at 10000 and 10001 samples are likely to be almost identical – so we would settle for *some* index with $p_i \approx \tau$.

For a non-computer science example, calculating the LD50 for a substance (the dose needed to kill half of the members of a specific population) is a noisy binary search problem with error probability that skyrockets close to the true answer.

Such considerations led to the noisy binary search model of Karp and Kleinberg [10], which we call MonotonicNBS: we have n coins whose unknown probabilities $p_i \in [0,1]$ are sorted in nondecreasing order. We can flip coin i to see heads with probability p_i. The goal is to find any coin i with nonempty $[p_i, p_{i+1}] \cap (\tau - \varepsilon, \tau + \varepsilon)$ (See Figure 1 for a graphical representation). This model subsumes FixedNoiseNBS (where $p_i = \frac{1}{2} - \varepsilon$ for $i \le i^*$ and $\frac{1}{2} + \varepsilon$ otherwise) and of course regular binary search (where $p_i \in \{0,1\}$). Throughout this paper we will suppose that τ is a constant bounded away from $\{0,1\}$, n grows to ∞, and ε and the desired failure probability δ may be constant or may approach 0 as $n \to \infty$.

The naive solution to MonotonicNBS is binary search with repetition: we do regular binary search, but repeat each query enough times to have $\frac{\delta}{\log n}$ failure probability if $p_i \notin [\tau - \varepsilon, \tau + \varepsilon]$. This gives sample complexity $O(\frac{1}{\varepsilon^2} \log n \log \frac{\log n}{\delta})$. In [10] it was shown that this extra $\log \log n$ term is unnecessary, giving two algorithms that each have sample complexity

$$O(\frac{1}{\varepsilon^2} \log n \log \frac{1}{\delta}).$$

In this paper, we show how to improve this bound. We show upper and lower bounds that achieve the tight constant on $\log n$, and reduce the $\log \frac{1}{\delta}$ dependence from multiplicative to additive. Table 1 compares our result to existing methods for MonotonicNBS.

On Studying Constants. When analyzing sublinear algorithms, and trying to remove $\log \log n$ factors in query complexity, constant factors really matter. The proofs in [10] are not careful with constants, but the algorithms themselves inherently lose constants. Our best estimate is that one algorithm "improves" upon naive repetition by a factor of $\frac{\log \log_2 n}{31}$, and the other by $\frac{\log \log_2 n}{2000}$ (where log is the natural log). Neither is an improvement for any n that will ever be practical – the better algorithm is only an improvement for $n > 2^{e^{31}} \approx 10^{10^{11}}$. By studying constants, we are forced to design an algorithm that (as we shall see) gives improvements for practical values of n. We give further discussion of the value of studying constants in Section 2.2.

Connection to Communication. Noisy binary search is intimately connected to the asymmetric binary channel, i.e., the binary channel that can choose between sending 1 with probability $\tau - \varepsilon$ or with probability $\tau + \varepsilon$. If each $p_i \in \{\tau \pm \varepsilon\}$, then noisy binary search needs to reveal the $\log_2 n$-bit i^* through such a channel; queries below i^* are 1 with probability $\tau - \varepsilon$ and those above i^* are 1 with probability $\tau + \varepsilon$. The natural target sample complexity is therefore $\frac{1}{C_{\tau,\varepsilon}} \log_2 n$, where $C_{\tau,\varepsilon}$ is the *information capacity* of the asymmetric binary channel:

$$C_{\tau,\varepsilon} := \max_q H((1-q)(\tau-\varepsilon) + q(\tau+\varepsilon)) - (1-q)H(\tau-\varepsilon) - qH(\tau+\varepsilon) \tag{1}$$

where $H(p)$ is the binary entropy function. For $\tau = \frac{1}{2}$, the maximum is at $q = \frac{1}{2}$ and this is just $C_{\frac{1}{2},\varepsilon} = 1 - H(\frac{1}{2} - \varepsilon)$, the capacity of the binary *symmetric* channel with error probability $\frac{1}{2} - \varepsilon$. For $\tau \neq \frac{1}{2}$, the information obtained from $\tau - \varepsilon$ and $\tau + \varepsilon$ probability coins is not the same, so the capacity is achieved by getting $\tau + \varepsilon$ coins with some probability q different from $1/2$; it satisfies $C_{\tau,\varepsilon} \approx \frac{\varepsilon^2}{2\tau(1-\tau)\log 2}$ for fixed τ as $\varepsilon \to 0$.

Our results. Our main result is the following:

▶ **Theorem 1** (Upper bound). *Let $0 < \tau < 1$ be a constant. Consider any parameters $0 < \varepsilon, \delta < 1/2$ with $0 < \varepsilon < \min(\tau, 1-\tau)/2$. On any* MonotonicNBS$(\tau, \varepsilon)$ *input, the algorithm* BayesianScreeningSearch *uses at most*

$$\frac{1}{C_{\tau,\varepsilon}}(\log_2 n + O(\log^{2/3} n \log^{1/3} \frac{1}{\delta} + \log \frac{1}{\delta}))$$

queries and succeeds with probability $1 - \delta$.

Unlike [2, 1, 5, 15, 9], our results apply to MonotonicNBS, not just FixedNoiseNBS, so they do not restrict the value of p_i and handle $\tau \neq \frac{1}{2}$. Unlike [10], we achieve good constant factors, high-probability results, and a better scaling with the target τ. In particular, [10] scales multiplicatively rather than additively with $O(\log \frac{1}{\delta})$; and it uses a reduction that incurs a constant-factor *loss* for targets $\tau \neq \frac{1}{2}$, while Theorem 1 scales with $\Theta(\tau(1-\tau))$ so *improves* for $\tau \neq \frac{1}{2}$.

Using Shannon's strong converse theorem, we show that the dependence on n is tight: for $\varepsilon \gg n^{-1/4}$, any algorithm must sometimes use $(1 - o(1))\frac{1}{C_{\tau,\varepsilon}} \log_2 n$ queries; in fact, it must use this many queries with nearly $1 - \delta$ probability.

▶ **Theorem 2** (Strong converse). *Any* MonotonicNBS(τ, ε) *algorithm that succeeds with $1 - \delta$ probability on inputs with all $p_i \in \{\tau \pm \varepsilon\}$ must have at least a $1 - \delta - O(\frac{1}{\gamma^2 n \varepsilon^4})$ chance of using at least*

$$(1-\gamma)\frac{1}{C_{\tau,\varepsilon}} \log_2 n$$

queries, for any $\gamma > 0$.

For $\tau = \frac{1}{2}$, this is also a lower bound for FixedNoiseNBS. Thus Theorem 2 gives a new *worst-case* lower bound for FixedNoiseNBS, which is a $\frac{1}{1-\delta}$ factor larger than the lower bound for *expected* query complexity achieved in prior work [2, 1, 5, 9].

For $\tau \neq \frac{1}{2}$, our results are the first ones connecting noisy binary search to $C_{\tau,\varepsilon}$, the information capacity of the binary asymmetric channel.

Our results: expected queries. For constant δ, one can get a better bound for the expected number of queries in a simple way: only run the algorithm with probability $1 - (1 - \frac{1}{\log n})\delta$, and otherwise output the wrong answer from zero queries. This saves essentially a $1 - \delta$ factor in queries, which for constant δ is nontrivial:

▶ **Corollary 3** (Upper bound: expected queries). *Under the same conditions as Theorem 1 and for any* MonotonicNBS(τ, ε) *input, algorithm* SillyBayesianScreeningSearch *uses*

$$\frac{1-\delta}{C_{\tau,\varepsilon}}(\log_2 n + O(\log^{2/3} n \log^{1/3} \frac{\log n}{\delta} + \log \frac{1}{\delta}))$$

queries in expectation and succeeds with probability $1 - \delta$.

This $1 - \delta$ savings is essentially the best possible. Our strong converse (Theorem 2) already implies this, if $\varepsilon \gg n^{-1/4}$; but using Fano's inequality, the optimality is true in general:

▶ **Theorem 4** (Weak converse). *Any* MonotonicNBS(τ, ε) *algorithm that succeeds with* $1 - \delta$ *probability on inputs with all* $p_i \in \{\tau \pm \varepsilon\}$ *must use*

$$(1-\delta)\frac{\log_2(n-2) - 1}{C_{\tau,\varepsilon}}$$

queries in expectation.

Theorem 4 was essentially shown in [2], which proved the $\tau = \frac{1}{2}$ case (by giving hardness for FixedNoiseNBS).

Our results: experiments. In Section A we compare our approach to naive repetition and the [10] algorithms. We find, for $n \geq 10^3$ and $\varepsilon = .1$, that our approach outperforms naive repetition, which outperforms both [10] algorithms. For $n = 10^9$, our approach uses $2.3\times$ fewer samples than naive repetition.

1.1 Algorithm Overview

We now describe our noisy binary search algorithm in the case of $\tau = \frac{1}{2}$ and $\delta > 1/n^{o(1)}$.

Bayesian start. The natural choice for a "hard" instance is when $p_i \in \{\tau \pm \varepsilon\}$ (though we will see that having multiple right answers is also hard in a different way), so the algorithm must find the transition location i^*, and information theoretic arguments show $\frac{1}{C_{\tau,\varepsilon}} \log_2 n$ queries are necessary. To avoid losing a constant factor in sample complexity, the algorithm essentially must spend most of its time running the Bayesian algorithm. This algorithm starts with a uniform prior over which interval crosses τ, makes the maximally informative query, updates its posterior, and repeats. When $\tau = \frac{1}{2}$, the maximally informative query is the median under the posterior, and the Bayesian update is to multiply intervals on one side of the query by $1 + 2\varepsilon$ and the other side by $1 - 2\varepsilon$. This algorithm, BayesLearn, is given in Algorithm 1; the algorithm for general τ is given in Section 4.

As a technical side note, the discrete nature of the problem introduces a bit of subtlety. Note that MonotonicNBS flips coins i but returns an *interval* between coins that should be good:

▶ **Definition 5.** *We say that an interval* $[i, i+1]$ *is* (τ, ε)-good *if* $[p_i, p_{i+1}] \cap (\tau - \varepsilon, \tau + \varepsilon)$ *is nonempty.*

Precisely, our version of the Bayesian algorithm is as follows: we start with a uniform prior over intervals. The median of our posterior can be viewed as a fractional coin, and we flip the nearest actual coin but update our posterior as if we flipped the fractional coin. So, for example, suppose the median is 4.7 ($.7 * w(5) + \sum_{i=1}^{4} w(i) = .5$). We flip coin 5, and if it comes out 0, that suggests the true threshold is probably above 5. We then scale up our posterior on all intervals above 5 by $1 + 2\varepsilon$; scale down intervals below 4 by $1 - 2\varepsilon$; and scale the weight on interval $[4, 5]$ by $.3(1 + 2\varepsilon) + .7(1 - 2\varepsilon)$. This new posterior is still a distribution that sums to 1.

Algorithm 1 Bayesian learner in $\tau = \frac{1}{2}$ case. Flips M coins and returns M intervals.

```
    Input A set of $n$ queryable coins, update size $\varepsilon$, number of steps $M$.
    Output A list of $M$ intervals queried.
1: procedure BayesLearn(coins, $\varepsilon$, $M$)
2:     $n \leftarrow |\text{coins}|$
3:     $w_1 \leftarrow \text{uniform}([n-1])$                    ▷ Prior distribution over intervals
4:     $L \leftarrow \{\}$
5:     for $i \in [M]$ do
6:         $j_i \leftarrow$ median interval of $w_i$
7:         $x_i \leftarrow$ either $j_i$ or $j_i + 1$, whichever is closer to the median
8:         append $j_i$ to $L$
9:         $y_i \leftarrow$ flip coin $x_i$                              ▷ 1 with probability $p_{x_i}$
10:        $w_{i+1}(x) \leftarrow \begin{cases} w_i(x)\cdot(1-2\varepsilon(-1)^{y_i}) & \text{if } x < j_i \\ w_i(x)\cdot(1+2\varepsilon(-1)^{y_i}) & \text{if } x > j_i \\ \text{remainder so } w_{i+1} \text{ sums to } 1 & \text{if } x = j_i \end{cases}$
11:    return $L$
```

Using the result. After running the Bayesian algorithm for most of our query budget, we need to output an answer. The question becomes: how can we take the transcript of the Bayesian algorithm and extract a useful worst-case frequentist guarantee? We need the algorithm to work for *all* monotonic p, which can have values very different than $\tau \pm \varepsilon$.

In the prior work achieving tight constants for FixedNoiseNBS [2, 5], because the p_i are guaranteed to be $\frac{1}{2} \pm \varepsilon$, the analysis can show that the weight of the single "good" interval grows in expectation at each step. By a Hoeffding bound, after the desired number of iterations the "good" interval has more weight than every other interval combined, so it can be easily selected. But that property is not true for the more general p_i of MonotonicNBS: if many p_i are $\frac{1}{2} \pm 0.6\varepsilon$, the Bayesian algorithm will wander somewhat too slowly through these samples without growing any single interval by the desired amount.

However, in such cases the Bayesian algorithm is spending a lot of time among good intervals. This holds in general. Our key lemma shows that, if we run BayesLearn for $1 + O(\gamma)$ times the information theoretic bound $\frac{1}{C_{\tau,\varepsilon}} \log_2 n$, a γ fraction of the intervals it visits are (τ, ε)-good:

▶ **Lemma 6** (Bayesian performance). *Consider any $0 < \varepsilon, \tau, \delta, \gamma < 1$ with $\gamma \le \frac{1}{7}$, $\varepsilon < \min(\tau, 1-\tau)/2$, and let L be the list of intervals returned by* BayesLearn, *when run for*

$$\frac{1 + O(\gamma)}{C_{\tau,\varepsilon}} \cdot \left(\log_2 n + O(\sqrt{\log n \log \frac{1}{\delta}} + \log \frac{1}{\delta}) \right)$$

iterations on an MonotonicNBS *instance. With probability $1 - \delta$, at least a γ fraction of the intervals in L are (τ, ε)-good.*

2 Proofs of Statements

By considering the γ-quantiles of the returned list, we reduce n to $\frac{1}{\gamma}$. We can now run a less efficient noisy binary search algorithm on this small subproblem. There are some complications, as the solution to the new noisy binary search could correspond to a larger interval than two adjacent coins. To deal with this, we run BayesLearn with $\varepsilon' = (1-o(1))\varepsilon$, which lets us test our candidate answers.

Technical comparison of techniques. How we leverage the bayesian learner is the main technical difference between our upper bound and that of prior work [10, 2, 5]. As described above, the situation is rather simpler for FixedNoiseNBS. For MonotonicNBS, [10] instead used conservative updates in their multiplicative weights algorithm: rather than the true Bayesian update $1 \pm 2\varepsilon$, it multiplies by about $1 \pm \frac{3}{5}\varepsilon$. This necessarily loses a constant factor, but ensures that either the median interval queried or the last interval queried is good. This property is not true for the true Bayesian algorithm with sharp constant.

2.1 Related Work

The FixedNoiseNBS version of noisy binary search, where $\tau = \frac{1}{2}$ and $p_i \in \{\frac{1}{2} \pm \varepsilon\}$, was posed by Burnashev and Zigangirov [2], who showed how to achieve

$$\frac{1}{C_{\frac{1}{2},\varepsilon}} \left(\log_2 n + \log_2 \frac{1}{\delta} + \log_2 \frac{1+2\varepsilon}{1-2\varepsilon} \right)$$

expected queries (in Russian; see [15] for an English proof). Essentially the same [2] algorithm for FixedNoiseNBS was rediscovered in [1]. Some bugs with the [1] proof were discovered and fixed in [5], as well as an analysis of a variant of the algorithm for *worst-case* sample complexity

$$\frac{1}{C_{\frac{1}{2},\varepsilon}} \left(\log_2 n + O(\sqrt{\log n \log \frac{1}{\delta}}) + O(\log \frac{1}{\delta}) \right).$$

For $1 \ll \log \frac{1}{\delta} \ll \log n$, Gu and Xu [9] showed black-box improvements for other δ. If δ is constant, they output $\perp$ with probability $\delta - \frac{1}{\log n}$, and otherwise run the [5] algorithm with $\delta' = \frac{1}{\log n}$. On the other hand, for $\delta = n^{-\Omega(1)}$, repeatedly running [5] with $\delta' = \frac{1}{\log n}$ and checking the result gives improvements:

$$(1+o(1)) \left(\frac{1-\delta}{C_{\frac{1}{2},\varepsilon}} \log_2 n + \frac{\log \frac{1}{\delta}}{\varepsilon \log \frac{1+2\varepsilon}{1-2\varepsilon}} \right)$$

For $\varepsilon \ll 1$, this is a factor 2 improvement on the constant factor on $\log \frac{1}{\delta}$. Moreover, [9] shows that this bound is sharp in both n and δ.

Our version of noisy binary search, MonotonicNBS, was first posed by Karp and Kleinberg [10]. They gave two algorithms, based on recursive backtracking and multiplicative weights respectively, that take $O(\frac{1}{\varepsilon^2} \log n)$ queries for constant δ, which they showed is within constant factors of optimal for constant τ, δ. Unfortunately, the constant factors make both algorithms worse than the naive repetition algorithm for any reasonable n (see Table 1 and Section A).

Other models. There are many different variations for noisy binary search (see [12] for a survey of older work on the subject). Emamjomeh-Zadeh, Kempe, and Singhal [7] solve an extension of FixedNoiseNBS from the line to graphs. This result was improved and simplied by Dereniowski, Tiegel, Uznański and Wolleb-Graf [4], which was in later improved and simplified by Dereniowski, Łukasiewicz, and Uznański [5]. Nowak developed a different generalization of FixedNoiseNBS to general hypothesis classes [11]. Waeber, Frazier, and Henderson [14] investigates a continuous variant of FixedNoiseNBS, where the target is a point in the real interval $[0, 1]$, and show that the Bayesian algorithm converges geometrically (the ideal convergence up to constant factors).

To our knowledge, [10] is the only previous work that handles a setting like MonotonicNBS where the "true" coin may be indistinguishable from nearby coins, and the goal is just to find a sufficiently good answer.

Applications. Noisy binary search is also used as a subroutine in other algorithms. For instance in [13] it is used for group testing, and in Crume [3] as a replacement for git-bisect under unreliable tests. Both implementations were based on the multiplicative weights algorithm of Karp and Kleinberg [10].

2.2 Why constants?

There is a tendency in theoretical computer science to regard constant factors as unimportant. But theorists care about constants in many situations, such as approximation ratios or rates of codes, and we believe that the query complexity of sublinear algorithms is another situation where they should be considered.

In general, the arguments for ignoring constants in time complexity hold with much less force for query complexity. The constant for time complexity is highly dependent on the machine architecture, which changes over time (e.g., the relative cost of addition and multiplication). Moreover, these hardware improvements mitigate the cost of poor constants. But the number of queries is a mathematical value, and the cost of queries (which may be, e.g., blood tests or running a giant test suite) does not clearly decrease with time.

The question should be: does theoretical study of constant factors lead to algorithmic insights necessary for more practical algorithms? Our paper shows that it does. By considering constants, we are forced to find a more efficient way of translating the Bayesian algorithm into one with frequentist guarantees (via Lemma 6). The constants lost in the previous attempt at this (in [10]) mean that it is worse than the naive method until $n > 10^{10^{11}}$.

It should not be surprising that a simple method that loses an $O(\log \log n)$ factor can beat an algorithm that loses "only" constants, for all practical values of n. The study of leading constants is a lens by which we found a new algorithm that actually outperforms the naive method for reasonable values of n (namely $n > 1000$).

3 Detailed Proof Sketch for Upper Bound

3.1 Key Lemma on Bayesian Learner

For this proof overview, we focus on the case of $\delta > n^{-o(1)}$ and target $\tau = \frac{1}{2}$, where BayesLearn queries the median of the posterior at each stage, and

$$C_{\tau,\varepsilon} = 1 - H(\frac{1}{2} + \varepsilon) = (\frac{1}{2} + \varepsilon) \log_2(1 + 2\varepsilon) + (\frac{1}{2} - \varepsilon) \log_2(1 - 2\varepsilon) \approx \frac{2\varepsilon^2}{\log 2}.$$

We give an overview of the proof of our key lemma in this case:

▶ **Lemma 6** (Bayesian performance). *Consider any $0 < \varepsilon, \tau, \delta, \gamma < 1$ with $\gamma \le \frac{1}{7}$, $\varepsilon < \min(\tau, 1-\tau)/2$, and let L be the list of intervals returned by* BayesLearn, *when run for*

$$\frac{1+O(\gamma)}{C_{\tau,\varepsilon}} \cdot \left(\log_2 n + O(\sqrt{\log n \log \frac{1}{\delta}} + \log \frac{1}{\delta})\right)$$

iterations on an MonotonicNBS *instance. With probability $1-\delta$, at least a γ fraction of the intervals in L are (τ,ε)-good.*

Let a be the "best answer", an interval that straddles the bias $\frac{1}{2}$. The algorithm keeps track of a distribution w on $[n-1]$; at each step i, it queries the median of the current distribution w_i, then multiplies the density on one side by $1+2\varepsilon$ and the other by $1-2\varepsilon$ to form w_{i+1}. We analyze the algorithm by looking at $\log_2 w(a)$.

At each step, the interval j we choose is either good (a valid answer) or bad (invalid). If it is *bad*, suppose the sampled coin x has probability $p_x \ge \frac{1}{2} + \varepsilon$. Then x is above a, so $w(a)$ multiplies by $1+2\varepsilon$ with probability p_x, and $1-2\varepsilon$ with probability $1-p_x$. Hence:

$$\mathbb{E}[\log_2 w_{i+1}(a) - \log_2 w_i(a)] = p_x \log_2(1+2\varepsilon) + (1-p_x)\log_2(1-2\varepsilon) \ge C_{\tau,\varepsilon}.$$

The case of $p_x \le \frac{1}{2} - \varepsilon$ is symmetric, giving the same bound. So every bad interval we select increases $\log_2 w(a)$ by $C_{\tau,\varepsilon}$ in expectation.

On the other hand, if the interval we select is *good*, $\log_2 w(a)$ may decrease in expectation. For example, if we query coin a and $\sum_{i=1}^{a-1} w(i) = \frac{1}{2}$, we could have

$$\mathbb{E}[\log_2 w_{i+1}(a) - \log_2 w_i(a)] = \frac{1}{2}\log_2(1-2\varepsilon) + \frac{1}{2}\log_2(1+2\varepsilon) \approx -\frac{2\varepsilon^2}{\log 2} \approx -C_{\tau,\varepsilon}$$

It turns out this is essentially the worst case, and in general the expected decrease in $\log_2 w(a)$ is no more than $5C_{\tau,\varepsilon}$ for any $\varepsilon < \frac{1}{2}\min(\tau, 1-\tau)$. As a result, the potential function

$$\log_2 w_i(a) - \gamma C_{\tau,\varepsilon} \cdot (\text{\# intervals chosen}) + 6C_{\tau,\varepsilon} \cdot (\text{\# good intervals chosen})$$

increases by at least $(1-\gamma)C_{\tau,\varepsilon}$ in expectation in each step i, regardless of where the median is in that step. This potential function starts at $-\log_2(n-1)$, so after $M = (1+2\gamma)\frac{1}{C_{\tau,\varepsilon}}\log_2 n$ steps it is at least $\Theta(\gamma)\log_2 n$ in expectation. An Azuma-Hoeffding bound shows that the value concentrates about this expectation, and in particular will be positive with $1-\delta$ probability. If so, since $\log_2 w_i(a) \le 0$ always, we have

$$6 \cdot (\text{\# good intervals chosen}) - \gamma(\text{\# intervals chosen}) \ge 0,$$

and hence a $\frac{\gamma}{6}$ fraction of chosen intervals are good.

This proves the key lemma: after $(1+O(\gamma))\frac{1}{C_{\tau,\varepsilon}}\log n$ steps of BayesLearn, a γ fraction of coins flipped are good with decent probability.

Targets $\tau \neq \frac{1}{2}$. When $\tau \neq \frac{1}{2}$, the maximum-information query is no longer the median coin, but a slightly different quantile $\frac{1}{2} \pm O(\frac{\varepsilon}{\tau(1-\tau)})$, and the Bayesian updates use more complicated factors. This choice is still capacity-achieving on bad intervals, i.e., the expected "information gain" is $\mathbb{E}[\log_2 w_i(a) - \log_2 w_{i+1}(a)] \ge C_{\tau,\varepsilon}$, and on good intervals the expected information loss is still at most $5C_{\tau,\varepsilon}$, so the proof structure works unchanged.

3.2 Rest of Upper Bound

Recall that in this overview we assume $\log\frac{1}{\delta} \ll \log n$. By Lemma 6, if we take all $\{\gamma, 2\gamma, \dots, \lfloor\frac{1}{\gamma}\rfloor\gamma\}$ quantiles of the list returned by BayesLearn, run with parameter $\varepsilon' = \varepsilon(1-\alpha)$ (where α is introduced so we can later test the bias of each coin), we get a size-$\frac{1}{\gamma}$ list containing at least one ε'-good interval. This ε' has $C_{\tau,\varepsilon'} = (1-O(\alpha))C_{\tau,\varepsilon}$. For any γ, we can just flip all of these coins $O(\frac{1}{\alpha^2\varepsilon^2}\log\frac{1}{\gamma\delta})$ times to find an ε-good one. This would give sample complexity

$$\underbrace{(1+O(\gamma))(1+O(\alpha))\frac{1}{C_{\tau,\varepsilon}}\left(\log_2 n + O(\sqrt{\log n \log\frac{1}{\delta}})\right)}_{\text{BayesLearn, Lemma 6}} + \underbrace{O(\frac{1}{\gamma}\cdot\frac{1}{\alpha^2\varepsilon^2}\log\frac{1}{\gamma\delta})}_{\text{Testing quantiles}} \tag{2}$$

which, by setting γ and α to $(\frac{\log\frac{1}{\delta}}{\log n})^{1/4}$, gives sample complexity

$$(1+O(\frac{\log\frac{1}{\delta}}{\log n})^{1/4})\frac{1}{C_{\tau,\varepsilon}}\log_2 n.$$

This is the desired sharp bound, within $(1+o(1))$ of optimal. One can do slightly better: the second stage is itself a noisy binary search question on $O(1/\gamma)$ coins, so by applying the algorithm recursively with $\gamma' = O(1)$ we can solve it on the size-$O(1/\gamma)$ list in $O(\frac{1}{(1-\alpha)C_{\tau,\varepsilon}}\log\frac{1}{\gamma\delta} + \frac{1}{\alpha^2\varepsilon^2}\log\frac{1}{\gamma\delta})$ queries. As we recurse on a much smaller list, the samples used are all lower order and we do not need to recurse more than once. However, the answer to the recursive call might not be a valid answer to the original problem. Regardless, one of the endpoints of the return call must be a valid answer, which we can test for. By optimizing the parameters, this improves the sample complexity to

$$(1+O(\frac{\log\frac{1}{\delta}}{\log n})^{1/3})\frac{1}{C_{\tau,\varepsilon}}\log_2 n,$$

giving Theorem 1.

4 Proof of Lemma 6

4.1 Definitions

Let $\{l, \dots, r\}$ be the set of good intervals. Let a be the maximum $i \in [n-1]$ such that $p_i \le \tau$. We also define the following functions:

$$C_{\tau,\varepsilon} = \max_q H((1-q)(\tau-\varepsilon)+q(\tau+\varepsilon)) - (1-q)H(\tau-\varepsilon) - qH(\tau+\varepsilon) \tag{3}$$

$$W(x) = \sum_{i\in[x]} w(i) \tag{4}$$

$$\Phi(w,L) = \log_2 w(a) + 6C_{\tau,\varepsilon}(|\{x \in L | x \in [l,r]\}| - \gamma|L|) \tag{5}$$

$$q = \arg\max_x H((1-x)(\tau-\varepsilon)+x(\tau+\varepsilon)) - (1-x)H(\tau-\varepsilon) - xH(\tau+\varepsilon) \tag{6}$$

$C_{\tau,\varepsilon}$ is the capacity of a (τ,ε)-BAC. We let q satisfy the equation which expresses the shared information between a sent and received message through a (τ,ε)-BAC. (See 12, 13 for explicit formulas for $C_{\tau,\varepsilon}, q$) If our prior were true – so the coins really were $\tau \pm \varepsilon$ – we would like to flip a $\tau+\varepsilon$ coin with probability q. This is achieved by selecting the q-quantile of our posterior, which is above the true threshold with probability q. If $\tau = \frac{1}{2}$, $q = \frac{1}{2}$ and we query the median; in general, we query the $q = \frac{1}{2} \pm O(\frac{\varepsilon}{\tau(1-\tau)})$ quantile.

Φ is a potential function that we will be analyzing. We also define:

$$d_{0,0} = \frac{1-\tau-\varepsilon}{1-\tau-(2q-1)\varepsilon}, d_{0,1} = \frac{1-\tau+\varepsilon}{1-\tau-(2q-1)\varepsilon}, d_{1,0} = \frac{\tau+\varepsilon}{\tau+(2q-1)\varepsilon}, d_{1,1} = \frac{\tau-\varepsilon}{\tau+(2q-1)\varepsilon} \tag{7}$$

for brevity. In terms of BayesLearn we can think of $d_{x,y}$ as "the multiplicative effect of a flip resulting in x ($1 = \text{Heads}, 0 = \text{Tails}$) on the density of an interval on side y ($1 = \text{Right}, 0 = \text{Left}$) of the flipped coin." When $\tau = \frac{1}{2}$, $d_{x,y} = 1 - 2\varepsilon(-1)^{x \oplus y}$.

Algorithm 2 Acts as a Bayesian learner for M iterations, returns a list of all the chosen intervals. Expressions for the $d_{x,y}$ values are given in (7).

```
1: procedure GETINTERVALFROMQUANTILE($w, q$)
2:     $i \leftarrow \min i \in [n]$ s.t. $W(i) \geq q$
3: procedure ROUNDINTERVALTOCOIN($i, w, q$)
4:     return $i$ if $\frac{q-W(i-1)}{w(i)} \leq q$ else $i+1$
5: procedure BAYESLEARN($\{c_i\}_{i=1}^n, n, \tau, \varepsilon, M$)
6:     $w_1 \leftarrow$ uniform($[n-1]$)
7:     Define $q$ as in (13)                                   ▷ The quantile we choose
8:     $L \leftarrow \{\}$
9:     for $i \in [M]$ do
10:        $j_i \leftarrow$ GETINTERVALFROMQUANTILE($w_i, q$)          ▷ The chosen interval
11:        $x_i \leftarrow$ ROUNDINTERVALTOCOIN($j_i, w_i, q$) ▷ The index of the coin we are going to flip
12:        append $j_i$ to $L$
13:        $y_i \leftarrow$ FLIP($c_{x_i}$)
14:        $w_{i+1} \leftarrow \begin{cases} w_i(x)d_{y_i,0} & \text{if } x \in \{1,\dots,j_i-1\} \\ d_{y_i,0}(q-W_i(j_i-1)) + d_{y_i,1}(W_i(j_i)-q) & \text{if } x = j_i \\ w(x)d_{y_i,1} & \text{if } x \in \{j_i+1,\dots,n-1\} \end{cases}$
       return $L$
```

▶ **Lemma 7.** *In* BayesLearn, $\mathbb{E}[\Phi_{t+1} - \Phi_t | y_t, y_{t-1}, \dots, y_1] \geq (1 - O(\gamma))C_{\tau,\varepsilon}$.

Proof. Φ is given by the sum of equations (8) and (9).

$$6C_{\tau,\varepsilon}(|\{j \in L | j \in [l,r]\}| - \gamma|L|) \tag{8}$$

$$\log_2 w(a) \tag{9}$$

Recall that in the tth round, j_t is the interval chosen, and x_t is index of the coin flipped. Let p be the probability c_{x_t} lands heads.

Bad queries. Suppose $j_t \notin [l,r]$. If $j_t > r$, then $p \geq \tau + \varepsilon$ and the expected change in (9) is

$$p \log_2 d_{1,0} + (1-p)\log_2 d_{0,0}$$

The first log is positive and the second log is negative, so this expression is minimized at $p = \tau + \varepsilon$, at which point some computation (Lemma 9) shows that it equals $C_{\tau,\varepsilon}$. Similarly, if $x_t < l$ then $p \leq \tau - \varepsilon$ and the expected change is

$$p \log_2 d_{1,1} + (1-p)\log_2 d_{0,1}$$

which is also at least $C_{\tau,\varepsilon}$ by Lemma 9. As $j_t \notin [l,r]$, the change in (8) is $-\gamma \cdot 6C_{\tau,\varepsilon}$. Therefore in this case the expected change in Φ is at least $(1-6\gamma)C_{\tau,\varepsilon}$.

Good queries. Suppose $j_t \in [l, r]$. The change in (8) is now $6C_{\tau,\varepsilon}(1-\gamma)$. But how much can (9) decrease in expectation? Suppose that $j_t \neq a$. Then the expected change is either

$$p \log_2 d_{1,0} + (1-p) \log_2 d_{0,0}$$

with $p \geq \tau$, or

$$p \log_2 d_{1,1} + (1-p) \log_2 d_{0,1}$$

with $p \leq \tau$.

As $d_{1,0} = \frac{\tau+\varepsilon}{\tau+(2q-1)\varepsilon} \geq \frac{1-\tau-\varepsilon}{1-\tau-(2q-1)\varepsilon} = d_{0,0}$ and $d_{1,1} = \frac{\tau-\varepsilon}{\tau+(2q-1)\varepsilon} \leq \frac{1-\tau+\varepsilon}{1-\tau-(2q-1)\varepsilon} = d_{0,1}$, both of these expressions are minimized when $p = \tau$. So the expected change in (9) is lower bounded by:

$$\min\left(\tau \log_2 d_{1,0} + (1-\tau) \log_2 d_{0,0}, \tau \log_2 d_{1,1} + (1-\tau) \log_2 d_{0,1}\right). \tag{10}$$

We note that

$$\begin{aligned}
\tau \log_2 d_{1,0} + (1-\tau) \log_2 d_{0,0} &= (\tau+\varepsilon) \log_2 d_{1,0} + (1-\tau-\varepsilon) \log_2 d_{0,0} - \varepsilon \log_2 d_{1,0} + \varepsilon \log_2 d_{0,0} \\
&= C_{\tau,\varepsilon} - \varepsilon \log_2 d_{1,0} + \varepsilon \log_2 d_{0,0} \\
&\geq C_{\tau,\varepsilon} - 3\varepsilon\left(\frac{\varepsilon}{\tau} + \frac{\varepsilon}{1-\tau}\right) \qquad \text{(Lemma 13)} \\
&= C_{\tau,\varepsilon} - \frac{3\varepsilon^2}{\tau(1-\tau)} \\
&\geq C_{\tau,\varepsilon} - (6 \log 2) C_{\tau,\varepsilon} \qquad \text{(Lemma 10)} \\
&\geq -5C_{\tau,\varepsilon}
\end{aligned}$$

a symmetric argument for lower bounding $\tau \log_2 d_{1,1} + (1-\tau) \log_2 d_{0,1}$ holds. Therefore, the change in (9) is lower bounded by $-5C_{\tau,\varepsilon}$.

Now suppose that $j_t = a$. Then for some $k \in [0, 1]$ the expected change in (9) is:

$$p \log_2(d_{1,0}k + d_{1,1}(1-k)) + (1-p) \log_2(d_{0,0}k + d_{0,1}(1-k))$$

If $k \leq q$ then we flip a so $p \leq \tau$. $d_{0,0}k + d_{0,1}(1-k) \geq d_{0,0}q + d_{0,1}(1-q) = 1$. Also $d_{1,0}k + d_{1,1}(1-k) \leq d_{1,0}q + d_{1,1}(1-q) = 1$. Therefore, this expression is minimized when $p = \tau$. By symmetry, when $k > q$ this expression is also minimized when $p = \tau$.

So the expected change in (9) is lower bounded by

$$\tau \log_2(d_{1,0}k + d_{1,1}(1-k)) + (1-\tau) \log_2(d_{0,0}k + d_{0,1}(1-k))$$

for some $k \in [0, 1]$. Taking the derivative with respect to k, we get

$$\tau \frac{d_{1,0} - d_{1,1}}{d_{1,1} + (d_{1,0} - d_{1,1})k} + (1-\tau) \frac{d_{0,0} - d_{0,1}}{d_{0,1} + (d_{0,0} - d_{0,1})k}$$

As $d_{1,1} < d_{1,0}$ and $d_{0,1} > d_{0,0}$, $\tau \frac{d_{1,0}-d_{1,1}}{d_{1,1}+(d_{1,0}-d_{1,1})k} > 0 > (1-\tau) \frac{d_{0,0}-d_{0,1}}{d_{0,1}+(d_{0,0}-d_{0,1})k}$. We note that as k increases, $\tau \frac{d_{1,0}-d_{1,1}}{d_{1,1}+(d_{1,0}-d_{1,1})k}$ decreases in magnitude, while $(1-\tau) \frac{d_{0,0}-d_{0,1}}{d_{0,1}+(d_{0,0}-d_{0,1})k}$ increases in magnitude. Therefore, the minimum value of the above expression is achieved when $k = 0$ or $k = 1$.

So the expected change in (9) is lower bounded by

$$\min(\tau \log_2 d_{1,0} + (1-\tau) \log_2 d_{0,0}, \tau \log_2 d_{1,1} + (1-\tau) \log_2 d_{0,1})$$

which is the same expression which we lower bounded for the $j_t \neq a$ case. Combining these two cases, when we are querying a good interval, the expected change is lower bounded by $6C_{\tau,\varepsilon}(1-\gamma) - 5C_{\tau,\varepsilon} = (1-6\gamma)C_{\tau,\varepsilon}$. Therefore $\mathbb{E}[\Phi_{t+1} - \Phi_t | y_t, y_{t-1}, \ldots, y_1] \geq (1 - O(\gamma))C_{\tau,\varepsilon}$. ◀

Now that we have the gain in our potential function, we can apply a stochastic domination argument with Freedman's inequality in order to get Lemma 6 (Appendix C).

5 Algorithm and Analysis

Algorithm 3 Noisy Binary Search. It recurses at most once.

```
    Input $n$ queryable coins $\{c_i\}_{i=1}^n$, update size $\varepsilon$, target $\tau$, failure probability $\delta$
    Output An interval which is $(\tau, \varepsilon)$-good
1:  procedure ReductionToGamma($\{c_i\}_{i=1}^n, n, \tau, \varepsilon, \delta, \gamma$)
2:      $L \leftarrow$ BayesLearn($\{c_i\}_{i=1}^n, n, \tau, \varepsilon, \frac{1+O(\gamma)}{C_{\tau,\varepsilon}}(\log_2 n + O(\sqrt{\log n \log \frac{1}{\delta}} + \log \frac{1}{\delta}))$)
3:      $R \leftarrow \{\}$
4:      for $i \in [\lfloor \frac{|L|}{\lceil \gamma |L| \rceil} \rfloor]$ do
5:          append $L_{\lceil \gamma |L| \rceil i}$ to $R$
        return Sorted(removeDuplicates($R$))
6:  procedure BayesianScreeningSearch($\{c_i\}_{i=1}^n, n, \tau, \varepsilon, \delta, \gamma = \frac{1}{7\log_2(n)}$)
7:      $\varepsilon' = \varepsilon \cdot \max(1 - \sqrt[3]{\log_n(1/\delta)}, \frac{2}{3})$
8:      $R \leftarrow$ ReductionToGamma($\{c_i\}_{i=1}^n, n, \tau, \varepsilon', \delta/3, \frac{1}{3\log_2(n)}$)
9:      if $|R| > 7$ then
10:         $R \leftarrow [1] + R + [n]$                 ▷ Pad $R$ with the extremes of the initial problem.
11:         $i \leftarrow$ BayesianScreeningSearch($\{c_{R_i}\}_{i=1}^{|R|}, |R|, \tau, \varepsilon', \delta/3, \frac{1}{7}$)
12:         $\hat{p}_{R_i+1} \leftarrow$ estimate $p_{R_i+1}$ up to $\pm \frac{\varepsilon-\varepsilon'}{2}$ error with $\delta/3$ f.p.
13:         if $\hat{p}_{R_i+1} > \tau - \varepsilon + \frac{\varepsilon-\varepsilon'}{2}$ then
14:             return $R_i$
15:         else
16:             return $R_{i+1} - 1$
17:     else
18:         for $x \in R$ do
19:             $\hat{p}_{x+1} \leftarrow$ estimate $p_{x+1}$ up to $\pm \frac{\varepsilon-\varepsilon'}{2}$ error with $\delta/18$ f.p.
20:             if $\hat{p}_{x+1} > \tau - \varepsilon + \frac{\varepsilon-\varepsilon'}{2}$ then
21:                 return $x$
```

▶ **Theorem 1** (Upper bound). *Let $0 < \tau < 1$ be a constant. Consider any parameters $0 < \varepsilon, \delta < 1/2$ with $0 < \varepsilon < \min(\tau, 1-\tau)/2$. On any* MonotonicNBS$(\tau, \varepsilon)$ *input, the algorithm* BayesianScreeningSearch *uses at most*

$$\frac{1}{C_{\tau,\varepsilon}}(\log_2 n + O(\log^{2/3} n \log^{1/3} \frac{1}{\delta} + \log \frac{1}{\delta}))$$

queries and succeeds with probability $1 - \delta$.

Proof.

Correctness. Suppose that we run BayesianScreeningSearch on a MonotonicNBS instance with parameters $\{c_i\}_{i=1}^n, n, \tau, \varepsilon, \delta$. Also assume that all probabilistic stages succeed, meaning that ReductionToGamma, BayesianScreeningSearch, and our coin bias estimation all succeed. By a union bound, this occurs with probability $\geq 1 - \delta$.

As we pick every $\gamma|L|$th coin from L and L contains at least $\lceil \gamma |L| \rceil$ ε'-good intervals, R contains at least one ε'-good interval. Suppose that $|R| \leq 7$ and that R_i is the first ε'-good interval in R.

Then for all $j \in \{1, \ldots, i-1\}$, either R_j is an ε-good interval or it is not. If it is, then we have nothing to worry about outputting it. If it is not, then $p_{R_j+1} \leq \tau - \varepsilon$ (as if $p_{R_j+1} \geq \tau + \varepsilon$ then R_i is not ε-good), so $\hat{p}_{R_j+1} \leq \tau - \varepsilon + \frac{\varepsilon - \varepsilon'}{2}$. So we do not output any not ε-good interval before R_i. Once we reach R_i, $p_{R_i+1} > \tau - \varepsilon'$, so $\hat{p}_{R_i+1} > \tau - \varepsilon' - \frac{\varepsilon-\varepsilon'}{2} = \tau - \varepsilon + \frac{\varepsilon - \varepsilon'}{2}$ and we output R_i.

Now suppose that $|R| > 7$. As we recursively run BayesianScreeningSearch with $\gamma = 1/7$, we note that for the R in the recursive call R', $|R'| = \lfloor \frac{|L|}{\lceil \gamma |L| \rceil} \rfloor \leq \lfloor \frac{1}{\gamma} \rfloor = 7$, so $|R'| \leq 7$. By our work above, this means that the recursive call returns i such that $[p_{R_i}, p_{R_{i+1}}] \cap (\tau - \varepsilon', \tau + \varepsilon') \neq \emptyset$.

Either R_i or $R_{i+1} - 1$ is ε'-good, as if $p_{R_i+1} \leq \tau - \varepsilon'$ and $p_{R_{i+1}-1} \geq \tau + \varepsilon'$ then R must not contain any good intervals. The same logic as for the $|R| \leq 7$ case holds, and we have shown correctness.

Number of samples. Suppose that we run BayesianScreeningSearch with $\gamma = 1/7$. The ReductionToGamma call takes

$$\frac{1 + O(\gamma)}{C_{\tau,\varepsilon'}} \left(\log_2 n + O(\sqrt{\log n \log \frac{1}{\delta}} + \log \frac{1}{\delta}) \right) = \frac{1}{C_{\tau,\varepsilon}} O(\log n + \log \frac{1}{\delta})$$

samples. As we have $\gamma = 1/7$, $|R| \leq 7$ and we go through the second branch. Then the bias estimation takes $O(\frac{\tau(1-\tau)\log\frac{1}{\delta}}{(\varepsilon - (1 - \sqrt[3]{\log_n \frac{1}{\delta}})\varepsilon)^2}) = O(\frac{\tau(1-\tau)\log\frac{1}{\delta}}{(\varepsilon \sqrt[3]{\log_n \frac{1}{\delta}})^2}) = O(\frac{\log^{2/3} n \log^{1/3} \frac{1}{\delta}}{C_{\tau,\varepsilon}})$ samples, for overall $\frac{1}{C_{\tau,\varepsilon}} O(\log n + \log \frac{1}{\delta})$ samples.

Now consider the case $\gamma = \frac{1}{7 \log_2 n}$, and suppose that $1 - \sqrt[3]{\log_n(1/\delta)} \geq 2/3$. When $\gamma = O(1/\log(n)), \varepsilon' = \varepsilon * (1 - \sqrt[3]{\log_n(1/\delta)})$, we have $\frac{1+O(\gamma)}{C_{\tau,\varepsilon'}} = \frac{1+O(\frac{1}{\log n})}{C_{\tau,\varepsilon'}} = \frac{1}{C_{\tau,\varepsilon'}} = \frac{1}{(1 - O(\sqrt[3]{\log_n(1/\delta)})C_{\tau,\varepsilon}}$, by Lemma 12. So ReductionToGamma takes

$$\frac{1}{(1 - O(\sqrt[3]{\log_n(1/\delta)})C_{\tau,\varepsilon}} \cdot \left(\log_2 n + O(\sqrt{\log n \log \frac{1}{\delta}} + \log \frac{1}{\delta} \right) \\ = \frac{1}{C_{\tau,\varepsilon}} \cdot \left(\log_2 n + O(\log^{2/3} n \log^{1/3} \frac{1}{\delta} + \log \frac{1}{\delta}) \right)$$

samples.

If $|R| \leq 7$ we take the second branch and take $O(\frac{\log^{2/3} n \log^{1/3} \frac{1}{\delta}}{C_{\tau,\varepsilon}})$ more samples, which meets our bound. If $|R| > 7$ we take the first branch and recurse with $\gamma = 1/7$ and $n' = O(\log n)$, for $\frac{1}{C_{\tau,\varepsilon}} O(\log\log n + \log \frac{1}{\delta})$ samples.

As established previously, the bias estimation takes $O(\frac{\log^{2/3} n \log^{1/3} \frac{1}{\delta}}{C_{\tau,\varepsilon}})$ samples. Overall the algorithm takes

$$\frac{1}{C_{\tau,\varepsilon}} (\log_2 n + O(\log^{2/3} n \log^{1/3} \frac{1}{\delta} + \log \frac{1}{\delta}))$$

samples. In the case $1 - \sqrt[3]{\log_n(1/\delta)} < 2/3$, the $\frac{O(\log \frac{1}{\delta})}{C_{\tau,\varepsilon}}$ term dominates the rest, and the bound holds. ◀

Corollary 3 then follows by elementary analysis.

6 Lower Bounds

We achieve our lower bounds by a communication complexity reduction.

▶ **Lemma 8.** *Given any algorithm $\mathcal{A}$ which solves NBS for parameters (τ, ε) with sample budget m and failure probability δ, there exists a protocol that communicates over a discrete memoryless channel with capacity $C_{\tau,\varepsilon}$ with rate $R = \frac{\log_2(n-1)}{m}$ with failure probability δ.*

Now we can use lower bounds from information theory. By applying Shannon's Strong Converse Theorem, we get Theorem 2, and by applying Fano's inequality we get Theorem 4. See Appendix C for proofs.

7 Future Work

One interesting topic of research is *instance-dependent* noisy binary search. If an instance is much nicer than the worst case, say every coin has bias $\frac{1}{2} \pm \alpha$ for $\alpha \gg \varepsilon$, we would hope to get a $O(\frac{\log n}{\alpha^2})$ dependence, which BayesianScreeningSearch does not get. One could use an adaptive coin bias estimator to get some adaptivity, but the constants gotten from this will likely not be good.

Another open problem is attenuating the lower order terms in the upper bound for NBS. For realistic n, lower order terms such as $\sqrt{\log n}$, or even $\log \log n$ are not negligible compared to $\log n$, and influences the practical application of BayesianScreeningSearch, as seen in the experimental results where we spent 28% of our samples on the "lower order" recursive calls.

One conjectural algorithm for noisy binary search would be: run BayesLearn for $(1 + O(\gamma))OPT$ steps, then output the median of the last γOPT intervals chosen. This interpolates between the overall median (which loses a constant factor) and the final interval (which has a large probability of failure), and avoids the inefficiency of recursive calls.

References

1 Michael Ben-Or and Avinatan Hassidim. The bayesian learner is optimal for noisy binary search (and pretty good for quantum as well). In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 221–230, 2008. `doi:10.1109/FOCS.2008.58`.

2 Marat Valievich Burnashev and Kamil'Shamil'evich Zigangirov. An interval estimation problem for controlled observations. *Problemy Peredachi Informatsii*, 10(3):51–61, 1974.

3 Adam Crume. Robust binary search. `https://github.com/adamcrume/robust-binary-search`, 2020.

4 Dariusz Dereniowski, Daniel Graf, Stefan Tiegel, and Przemyslaw Uznanski. A framework for searching in graphs in the presence of errors. *CoRR*, abs/1804.02075, 2018. `arXiv:1804.02075`.

5 Dariusz Dereniowski, Aleksander Lukasiewicz, and Przemyslaw Uznanski. Noisy searching: simple, fast and correct. *CoRR*, abs/2107.05753, 2021. `arXiv:2107.05753`.

6 Ilias Diakonikolas, Themis Gouleakis, John Peebles, and Eric Price. Sample-optimal identity testing with high probability. In *45th International Colloquium on Automata, Languages, and Programming (ICALP 2018)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018.

7 Ehsan Emamjomeh-Zadeh, David Kempe, and Vikrant Singhal. Deterministic and probabilistic binary search in graphs. In *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '16, pages 519–532, New York, NY, USA, 2016. Association for Computing Machinery. `doi:10.1145/2897518.2897656`.

8 Robert G Gallager. *Information theory and reliable communication*, volume 2. Springer, 1968.

9 Yuzhou Gu and Yinzhan Xu. Optimal bounds for noisy sorting. *STOC*, 2023. `doi:10.48550/arXiv.2302.12440`.

10 Richard M. Karp and Robert Kleinberg. Noisy binary search and its applications. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, pages 881–890, USA, 2007. Society for Industrial and Applied Mathematics.

11 Robert Nowak. Noisy generalized binary search. In Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, volume 22. Curran Associates, Inc., 2009. URL: `https://proceedings.neurips.cc/paper/2009/file/556f391937dfd4398cbac35e050a2177-Paper.pdf`.

12 Andrzej Pelc. Searching games with errors—fifty years of coping with liars. *Theoretical Computer Science*, 270(1):71–109, 2002. `doi:10.1016/S0304-3975(01)00303-6`.

13 Bernard Teo and Jonathan Scarlett. Noisy adaptive group testing via noisy binary search. *IEEE Trans. Inf. Theory*, 68(5):3340–3353, 2022. `doi:10.1109/TIT.2022.3140604`.

14 Rolf Waeber, Peter I. Frazier, and Shane G. Henderson. Bisection search with noisy responses. *SIAM Journal on Control and Optimization*, 51(3):2261–2279, 2013. `doi:10.1137/120861898`.

15 Ziao Wang, Nadim Ghaddar, Banghua Zhu, and Lele Wang. Noisy sorting capacity. In *2022 IEEE International Symposium on Information Theory (ISIT)*, pages 2541–2546. IEEE, 2022.

A Figures and Experiments

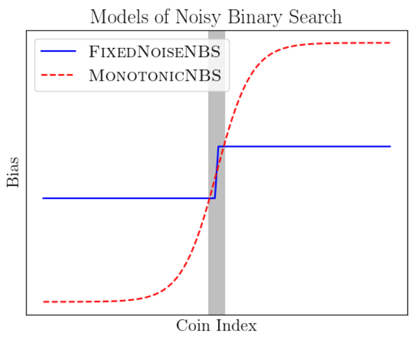

Figure 1 In FixedNoiseNBS, every coin is ε-far from the true i^* that must be found. We consider MonotonicNBS, where many coins may be close to the threshold and the goal is to find some good coin (the gray shaded region).

Applying NBS. To demonstrate the practicality of BayesianScreeningSearch we compare it to standard binary search with repetition (NaiveNBS) and the two algorithms of [10] (KKBacktracking and KKMultiplicativeWeights).

To fairly compare between these algorithms, we can't just use the descriptions given in [10], as the constants used in analysis are not optimized. We leverage BayesianScreeningSearch to address this. We tweak the listed algorithms so they take a sample budget as input which they allocate among all their stages. To estimate how large a budget is needed for algorithm $\mathcal{A}$ to perform well on distribution $\mathcal{D}$, we run BayesianScreeningSearch where when the ith coin is flipped we run $\mathcal{A}$ on some input drawn from $\mathcal{D}$, and return 1 if $\mathcal{A}$ succeeds and 0 if $\mathcal{A}$ fails. By setting $\tau = .8, .9$ and $\varepsilon = .05$, we get upper and lower bounds for how many samples is needed to get $\delta = .15$ failure probability.

Experiments. We compare results on 4 different problem distributions: Standard, Biased, Lopsided, and Wide.

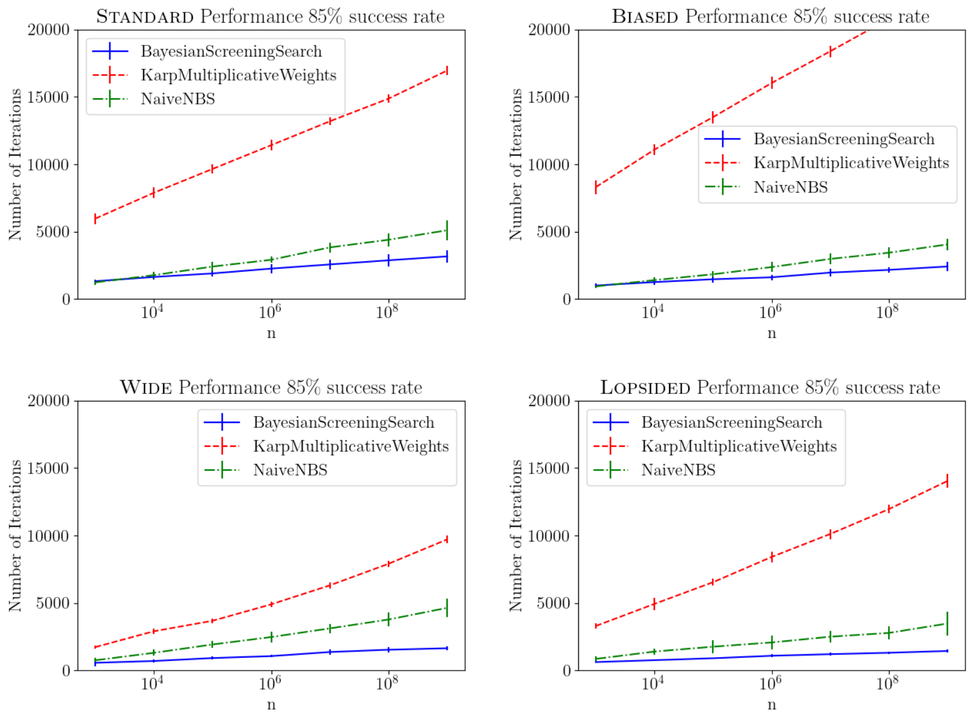

Figure 2 Performance of various MONOTONICNBS algorithms for listed distributions.

- STANDARD. $p_i \in \{\tau - \varepsilon, \tau + \varepsilon\}$, $\tau = \frac{1}{2}, \varepsilon = .1$, the transition interval chosen uniformly at random..
- BIASED. $p_i \in \{\tau - \varepsilon, \tau + \varepsilon\}, \tau = \frac{3}{4}, \varepsilon = .1$, the transition interval chosen uniformly at random.
- LOPSIDED. $p_i \in \{\tau - .6\varepsilon, \tau + \varepsilon\}$, $\tau = \frac{1}{2}, \varepsilon = .1$, the transition interval chosen uniformly at random..
- WIDE. we choose an interval (uniformly at random) of size $10 \log n$ that linearly interpolates between $\tau - \varepsilon$ and $\tau + \varepsilon$, and set the rest to be $p_i \in \{\tau - \varepsilon, \tau + \varepsilon\}, \tau = \frac{1}{2}, \varepsilon = .1$.

Results. We remark that KKBACKTRACKING performed markedly worse than the other algorithms, and so is not included in the figures. For reference, for STANDARD, $N = 1000$ KKBACKTRACKING required $m > 2.9 \times 10^6$ samples, while the other algorithms need $m < 6000$ samples (see Figure 2).

We find that KKMULTIPLICATIVEWEIGHTS is outperformed by NAIVENBS on all of these distributions. In contrast, BAYESIANSCREENINGSEARCH outperforms NAIVENBS for $n > 10^3$.

When $\tau \neq \frac{1}{2}$ the difference between BAYESIANSCREENINGSEARCH, NAIVENBS and the [10] algorithms increases. This is in line with our theory, as the first two perform better when τ is further from $\frac{1}{2}$, while the [10] algorithms reduce to the case $\tau = \frac{1}{2}$, losing a constant factor.

More details on the experiments are in the full version.

B Computations

This section gives the statements of some approximations used in the body of the paper. The proofs are in the full version.

We give explicit formulas for some functions used in this paper

$$z = 2^{\frac{H(\tau-\varepsilon)-H(\tau+\varepsilon)}{2\varepsilon}} \tag{11}$$

$$C_{\tau,\varepsilon} = \log_2(z+1) + \frac{\tau-\epsilon}{2\epsilon}H(\tau+\epsilon) - \frac{\tau+\epsilon}{2\epsilon}H(\tau-\epsilon) \tag{12}$$

$$q = \frac{(1-\tau+\varepsilon) - \frac{1}{1+z}}{2\varepsilon} \tag{13}$$

▶ **Lemma 9.**

$$C_{\tau,\varepsilon} = (\tau+\epsilon)\log_2\left(\frac{\tau+\epsilon}{\tau+(2q-1)\varepsilon}\right) + (1-\tau-\epsilon)\log_2\left(\frac{1-\tau-\epsilon}{1-\tau-(2q-1)\varepsilon}\right)$$

and

$$C_{\tau,\varepsilon} = (\tau-\epsilon)\log_2\left(\frac{\tau-\epsilon}{\tau+(2q-1)\varepsilon}\right) + (1-\tau+\epsilon)\log_2\left(\frac{1-\tau+\epsilon}{1-\tau-(2q-1)\varepsilon}\right)$$

▶ **Lemma 10.** *For* $\varepsilon \le \frac{1}{2}\min(\tau, 1-\tau)$,

$$\frac{1}{2\log 2}\frac{\varepsilon^2}{\tau(1-\tau)} \le C_{\tau,\varepsilon} \le \frac{1}{\log 2}\frac{\varepsilon^2}{\tau(1-\tau)}.$$

▶ **Lemma 11.** *For* $0 < \varepsilon \le \frac{1}{2}\min(\tau, 1-\tau)$,

$$|q - \frac{1}{2}| \le \frac{2\varepsilon}{\tau(1-\tau)}.$$

▶ **Lemma 12.**

$$C_{\tau,(1-o(1))\varepsilon} \ge (1-o(1))C_{\tau,\varepsilon}$$

▶ **Lemma 13.** *For* $\varepsilon < \frac{1}{2}\min(\tau, 1-\tau)$,

- $\log_2 d_{0,0} \ge -3\frac{\varepsilon}{1-\tau}$
- $\log_2 \frac{1}{d_{0,1}} \ge -3\frac{\varepsilon}{1-\tau}$
- $\log_2 \frac{1}{d_{1,0}} \ge -3\frac{\varepsilon}{\tau}$
- $\log_2 d_{1,1} \ge -3\frac{\varepsilon}{\tau}$

C Omitted Proofs

▶ **Lemma 6** (Bayesian performance). *Consider any* $0 < \varepsilon, \tau, \delta, \gamma < 1$ *with* $\gamma \le \frac{1}{7}$, $\varepsilon < \min(\tau, 1-\tau)/2$, *and let* L *be the list of intervals returned by* BayesLearn, *when run for*

$$\frac{1+O(\gamma)}{C_{\tau,\varepsilon}} \cdot \left(\log_2 n + O(\sqrt{\log n \log\frac{1}{\delta}} + \log\frac{1}{\delta})\right)$$

iterations on an MonotonicNBS *instance. With probability* $1-\delta$, *at least a* γ *fraction of the intervals in* L *are* (τ, ε)*-good.*

Proof of Lemma 6. In this proof we omit some of the routine calculations, see the full version for the complete proof.

Recall that Φ is given by the sum of equations (8) and (9).

$$6C_{\tau,\varepsilon}(|\{j \in L | j \in [l,r]\}| - \gamma|L|) \tag{8}$$

$$\log_2 w(a) \tag{9}$$

Reduction to $\Phi > 0$. First note that $\Phi_1 = -\log_2(n-1)$, as initially L is empty so (8) is 0, and we initialize w as uniform so $w(a) = \frac{1}{n-1}$. Next note that if (8) > 0, then

$$\begin{aligned} &6C_{\tau,\varepsilon}(|\{x \in L | x \in [l,r]\}| - \gamma|L|) > 0 \\ &\qquad\qquad |\{x \in L | x \in [l,r]\}| > \gamma|L| \end{aligned}$$

So there are strictly more than $\gamma|L|$ good intervals in L. Next note that (9) is ≤ 0 always, so $\Phi > 0 \implies$ (8) > 0. So it suffices to show that with δ failure probability $\Phi_{t+1} > 0$, where $t = \frac{1+O(\gamma)}{C_{\tau,\varepsilon}}(\log_2 n + O(\sqrt{\log n \log \frac{1}{\delta}} + \log \frac{1}{\delta}))$

Establishing a submartingale. By a stochastic domination argument, we can consider the worst case where all coins sampled have bias in $[\tau - \varepsilon, \tau + \varepsilon]$. Suppose that we are flipping a coin that has bias $p \le \tau - \varepsilon$. The potential function consists of two parts, one of which does not depend on the flip, and the other which does. We see that a tails increases the potential function, while a heads decreases it. So $p = \tau - \varepsilon$ is the worst case. The argument for when $p \ge \tau + \varepsilon$ is symmetric. We do not need to worry about how this affects future states as Lemma 7 conditions on the prior flips.

Let σ_i^2 be the variance of the difference random variables $\Phi_{i+1} - \Phi_i$, then we note that σ_i^2 is a Bernoulli random variable with parameter $p \in [\tau - \varepsilon, \tau + \varepsilon]$, that is scaled by at most a $\max(\log_2 d_{1,0} - \log_2 d_{0,0}, \log_2 d_{0,1} - \log_2 d_{1,1}) \lesssim \frac{\varepsilon}{\tau(1-\tau)}$ factor, therefore

$$\sigma_i^2 \lesssim p(1-p)\left(\frac{\varepsilon}{\tau(1-\tau)}\right)^2 \lesssim \tau(1-\tau)\left(\frac{\varepsilon}{\tau(1-\tau)}\right)^2 = \frac{\varepsilon^2}{\tau(1-\tau)} \lesssim C_{\tau,\varepsilon}$$

Where we use the fact that $\varepsilon \le \frac{\min(\tau,1-\tau)}{2}$. Therefore $\sigma_i^2 \lesssim C_{\tau,\varepsilon}$.

Freedman's inequality. For brevity let $g = (1 - O(\gamma))C_{\tau,\varepsilon}$, the lower bound given in Lemma 7.

$$\begin{aligned} \Pr[\Phi_{t+1} \le 0] &= \Pr[\Phi_{t+1} - \Phi_1 \le -\Phi_1] \\ &\le \exp\left(-\frac{2(-gt - \Phi_1)^2}{\sum_{i=1}^t \sigma_i^2 + O(\frac{\varepsilon}{\tau(1-\tau)})(gt + \Phi_1)}\right) \quad \text{Freedman's when } gt \ge -\Phi_1 \\ &\le \exp(-\frac{O(1)}{tC_{\tau,\varepsilon}} \cdot (g^2t^2 + 2gt\Phi_1 + \Phi_1^2)) \end{aligned}$$

Bounding this expression by δ, we get

$$g^2t^2 + (2g\Phi_1 - \log_{O(1)}(1/\delta)C_{\tau,\varepsilon})t + \Phi_1^2 \ge 0 \tag{14}$$

(14) is a quadratic with respect to t, and has a positive leading coefficient. Applying the quadratic formula, if we set

$$t \ge \frac{-\Phi_1}{g} + \frac{C_{\tau,\varepsilon}\log_{O(1)}(1/\delta)}{2g^2} + \frac{\sqrt{(\log_{O(1)}(1/\delta)C_{\tau,\varepsilon})^2 - 4g\Phi_1\log_{O(1)}(1/\delta)C_{\tau,\varepsilon}}}{2g^2} \tag{15}$$

then (14) holds. As $\Phi_1 = -\log_2(n-1), g = (1 - O(\gamma))C_{\tau,\varepsilon}$ we get that (15) is

$$\frac{1}{1 - O(\gamma)} \cdot \frac{1}{C_{\tau,\varepsilon}}(\log_2 n + O(\sqrt{\log n \log \frac{1}{\delta}} + \log \frac{1}{\delta}))$$

As $\frac{1}{1-O(\gamma)}$ is $1 + O(\gamma)$, our lemma holds. ◀

Algorithm 4 Noisy Binary Search that gets the optimal expected queries.

```
1: procedure SillyBayesianScreeningSearch($\{c_i\}_{i=1}^n, n, \tau, \varepsilon, \delta$)
2:     return $\begin{cases} \textsc{Random}([n-1]) & \text{w.p. } \delta - \delta/\log_2 n \\ \textsc{BayesianScreeningSearch}(\{c_i\}_{i=1}^n, n, \tau, \varepsilon, \frac{\delta}{\log(n)}) & \text{otherwise} \end{cases}$
```

Proof of Corollary 3. The failure probability of SillyBayesianScreeningSearch is $\leq \delta - \delta/\log_2 n + (1-\delta+\delta/\log_2 n)\delta/\log_2 n = \delta - \delta^2/\log_2 n + \delta^2/\log_2^2 n \leq \delta$. We use 0 samples with probability $\delta - \delta/\log_2 n$, and the expression in Theorem 1 with $\delta' = \delta/\log_2 n$, with probability $1-\delta+\delta/\log_2 n$. ◀

Proof of Lemma 8. Omitted, see the full version. ◀

▶ **Lemma 14** (Shannon's Strong Converse Theorem). *Over any discrete memoryless channel, for $R > C$*

$$P_e \geq 1 - \frac{K_1}{n(R-C)^2} - \exp(-K_2 n(R-C))$$

where P_e is the probability of error, K_1, K_2 are positive constants which depend on the channel, n is the input alphabet size, R is the rate of information, and C is the channel capacity [8].

Proof of Theorem 2. Let $\alpha = \frac{1}{1+\frac{K}{C_{\tau,\varepsilon}\sqrt{\beta(n-1)}}}$, for constant K to be determined later. Suppose that $\mathcal{A}$ uses at most $\alpha\frac{\log_2(n-1)}{C_{\tau,\varepsilon}}$ samples with probability at least $\delta+\beta$. Let $\mathcal{A}'$ be the algorithm that runs $\mathcal{A}$, but outputs a random answer if $\mathcal{A}$ uses more than $\alpha\frac{\log_2(n-1)}{C_{\tau,\varepsilon}}$ samples. $\mathcal{A}'$ fails only whenever $\mathcal{A}$ fails or uses more than $\alpha\frac{\log_2(n-1)}{C_{\tau,\varepsilon}}$ samples, so by a union bound $\mathcal{A}'$ has a failure probability of at most $1-\beta$. By Lemma 8 we can construct a protocol over a discrete memoryless channel with capacity $C_{\tau,\varepsilon}$ that communicates at rate $R = \frac{\log_2(n-1)}{\frac{\alpha\log_2(n-1)}{C_{\tau,\varepsilon}}} = \frac{C_{\tau,\varepsilon}}{\alpha} = \frac{C}{\alpha}$ with a failure probability of at most $1-\beta$.

By Lemma 14 we have that

$$\begin{aligned} 1-\beta &\geq 1 - \frac{K_1}{(n-1)(R-C)^2} - \exp(-K_2(n-1)(R-C)) \\ &= 1 - \frac{K_1}{(n-1)((1/\alpha-1)C)^2} - \exp(-(n-1)K_2((1/\alpha-1)C)) \\ &= 1 - \frac{K_1\beta}{K} - \exp(-\sqrt{\frac{n-1}{\beta}}K_2K) \\ &\geq 1 - \frac{\beta}{2} && \text{sufficiently large } K \text{ and } n \end{aligned}$$

which is a contradiction. Therefore with probability at least $1-\delta-\beta$ $\mathcal{A}$ uses $\frac{1}{1+\frac{K}{\sqrt{\beta(n-1)}(C_{\tau,\varepsilon})}}\frac{1}{C_{\tau,\varepsilon}}\log_2 n$ samples. ◀

Proof of Theorem 4. Omitted, see the full version. ◀